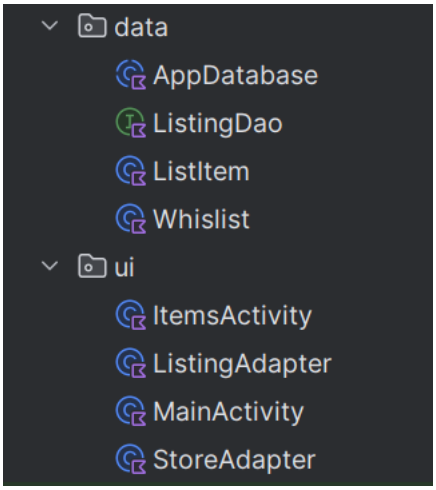
Kívánságlista alkalmazás – ewuxa6

Ez a dokumentum bemutatja az alkalmazás fő komponenseit, az adatréteget, a felhasználói felületet, a layoutokat, erőforrásokat, valamint az adatfolyamot és fejlesztési javaslatokat.

Bevezetés

Az alkalmazás célja, hogy a felhasználók egyszerűen kezelhessék a kívánságlistáikat és a hozzájuk tartozó termékeket. Az alábbiakban részletesen bemutatjuk a projekt struktúráját és komponenseit.



Adatréteg (data csomag)

AppDatabase

Szerep: Room adatbázis központi konfigurációja. Felelősség: entitások és DAO-k regisztrálása, singleton elérés.

ListingDao

Szerep: CRUD műveletek a listákra és listaelemekre. Annotációk: @Query, @Insert, @Update, @Delete.

ListItem

Szerep: Entitás a lista elemeihez (termékekhez). Mezők: id, listId, name, quantity, unit, isChecked.

```
@Entity(tableName = "shopping_items")
data class ListItem(
    @PrimaryKey(autoGenerate = true) val id: Int = 0,
    val listId: Int,
    val name: String,
    val category: String,
    val quantity: Int,
    val unit: String,
    val isBought: Boolean = false,
    val isUrgent: Boolean = false
)
```

Jelen verzióban a bekért adatokat nem korlátozom/ ellenőrzöm, illetve a kategória adat sem kerül sehol felhasználásra az alkalmzásban.

Whislist

Szerep: Entitás a listákhoz. Mezők: id, title, dueDate, status.

```
@Entity(tableName = "shopping_lists")
data class Whislist(
    @PrimaryKey(autoGenerate = true) val id: Int = 0,
    val name: String,
    val dueDate: String, // "yyyy-MM-dd" formátumban
    val totalItems: Int = 0,    // Kiszámolt érték
    val boughtItems: Int = 0    // Kiszámolt érték
)
```

További használt (számolt) állapotok:

Üres: tatolItems = 0
Kész: totalItems = boughtItems !=0
Lejáró/lejárt: dueDate < ma + 3 nap

UI réteg (ui csomag)

MainActivity

Szerep: Főképernyő, ahol a felhasználó összes listáját látja.

Funkciók:

Lista betöltése: Az alkalmazás indításakor a Room adatbázisból betölti az összes elérhető kívánságlistát LiveData vagy Flow segítségével, így a UI automatikusan frissül. A listák sorrendje előre meghatározott, priorizált, a sorrend azonal változik ha módosítást hajtunk végre.
Szűrés: A felhasználó státusz alapján szűrheti a listákat (pl. aktív, teljesített).
Új lista hozzáadása: Dialógus vagy külön képernyő megnyitása, ahol megadható a lista címe, határideje.
Törlés: A kis kuka ikonra koppintással törölhető egy lista, ami DAO-n keresztül frissíti az adatbázist.

Használt függvények:

- **onCreate:** UI és lokalizáció inicializálása, RecyclerView és „Új lista” gomb bekötése.
- **onResume:** Adatok újratöltése visszatéréskor.
- **refreshListData:** Listák megfigyelése, tételek számlálása, statisztikák előkészítése.

```
private fun refreshListData() {
    database.shoppingDao().getAllLists().observe( owner = this) { lists ->
        lifecycleScope.launch {
            val updatedLists = lists.map { list ->
                val items = database.shoppingDao().getItemsForListSync( listId = list.id)
                list.copy(
                    totalItems = items.size,
                    boughtItems = items.count { it.isBought }
                )
            }
            updateStatistics(updatedLists) // Statisztikai sor frissítése
        }
    }
}
```

- **updateStatistics:** Statisztika számítása/kiírása, szűrőgombok bekötése, aktuális szűrő alkalmazása.

```
private fun updateStatistics(updatedLists: List<Whislist>) {
    this.allLists = updatedLists

    val totalCount = updatedLists.size
    val emptyCount = updatedLists.count { it.totalItems == 0 }
    val inProgressCount = updatedLists.count { it.totalItems > 0 && it.boughtItems < it.totalItems }
    val doneCount = updatedLists.count { it.totalItems > 0 && it.boughtItems == it.totalItems }

    // Sürgős számolása (nem üres és nem kész listák)
    val urgentCount = updatedLists.count { isUrgent( list = it) }

    // UI elemek szövegének beállítása
    binding.statTotal.text = "📊 Összes: $totalCount (Szűrő törlése)"
    binding.statEmpty.text = "⚪ Üres: $emptyCount"
    binding.statInProgress.text = "🔵 Folyamatban: $inProgressCount"
    binding.statUrgent.text = "🔴 Sürgős: $urgentCount"
    binding.statDone.text = "⚪ Kész: $doneCount"

    // Kattintási események szűréshez
    binding.statTotal.setOnClickListener { applyFilter( filter = "ALL") }
    binding.statEmpty.setOnClickListener { applyFilter( filter = "EMPTY") }
    binding.statInProgress.setOnClickListener { applyFilter( filter = "PROGRESS") }
    binding.statUrgent.setOnClickListener { applyFilter( filter = "URGENT") }
    binding.statDone.setOnClickListener { applyFilter( filter = "DONE") }

    applyFilter(currentFilter)
}
```

- **isUrgent:** Határidő alapján sürgősség eldöntése (lejárt vagy 3 napon belüli).
- **applyFilter:** A logika szerinti szűrés és adapter frissítése, vizuális visszajelzés beállítása.

```
private fun applyFilter(filter: String) {
    currentFilter = filter
    val filteredList = when (filter) {
        "EMPTY" -> allLists.filter { it.totalItems == 0 }
        "PROGRESS" -> allLists.filter { it.totalItems > 0 && it.boughtItems < it.totalItems }
        "DONE" -> allLists.filter { it.totalItems > 0 && it.boughtItems == it.totalItems }
        "URGENT" -> allLists.filter { isUrgent( list = it) }
        else -> allLists
    }

    (binding.rvStores.adapter as StoreAdapter).setData(filteredList)

    updateFilterUI(filter)
}
```

- **setupRecyclerView:** Adapter és layout manager beállítása, törlés/megnyitás callbackek.
- **handleListClick:** Üres+lejárt eset kezelése párbeszéddel; különben megnyitás.
- **isPastDue:** Dátum lejártságának ellenőrzése.
- **openItemsActivity:** Navigáció a listaelemek képernyőre, extra adatokkal.
- **showAddStoreDialog:** Új lista hozzáadó dialógus, dátumválasztóval, DB-inzert.
- **updateFilterUI:** Szűrő vizuális státuszának (félkövér/szín) frissítése.

ItemsActivity

Elemek hozzáadása: Új termék felvétele név, mennyiség és mértékegység megadásával. A tételt fontossá lehet tenni, ilynekor az elemet mutató kártya kinézete is megváltozik (pirosan van kiemelve).
Törlés: Egy elem törlése a kuka ikonra való koppintással.
Pipálás: Checkbox segítségével jelölhető, hogy az adott termék beszerzésre került. Ez frissíti az isChecked mezőt az adatbázisban. Amennyiben egy listának minden eleme ki lett pipálva maga a lista státusza is kész lessz.
Visszalépés: Navigáció a főképernyőre, megőrizve az állapotot.

Használt függvények:

- **onCreate:** UI és adatforrás inicializálása, LiveData megfigyelés és „Új tétel” gomb bekötése.
- **setupRecyclerView:** Rácsos elrendezés (2 oszlop) beállítása és adapter hozzárendelése.
- **createAdapter:** Adapter létrehozása törlés, státuszváltás és szerkesztés callbackekkel.
- **showItemDialog:** Tétel felvétel/szerkesztés űrlappal; mentéskor DB beszúrás/frissítés.

Listing Adapter

RecyclerView adapter, amely a **bevásárlólista tételeit** jeleníti meg és kezeli. Három callbacket kap a konstruktorban, hogy az UI-ból indított események (törlés, státuszváltás, szerkesztés) azonnal a hívó rétegben (Activity/Fragment) végrehajthatók legyenek:

- onDelete(ListItem): adott tétel törlése.
- onStatusChange(ListItem): tétel állapotának (pl. megvett/pipált) frissítése.
- onEditClick(ListItem): tétel szerkesztő párbeszéd megnyitása.

Az adapter a tételeket **vizuálisan** is differenciálja (sürgős/piros kiemelés, megvett/áthúzott, halványítás), és **rendezi** őket a megjelenítés előtt (nem megvett → sürgős → név szerinti).

Store Adapter

RecyclerView adapter, amely a **kívánságlistákat** (Whislist) jeleníti meg kártyákon. Két callbacket vár a hívótól:

- **onCreateViewHolder(parent: ViewGroup, viewType: Int): StoreViewHolder**Felfújja az item_list layoutot, és visszaad egy új StoreViewHolder-t. Előkészíti a kártyanézetet az adatkötéshez.
- **onBindViewHolder(holder: StoreViewHolder, position: Int)**A megadott Whislist példány adatait a nézetekre köti (név, progressz), kiszámolja a lejárt/sürgős/üres/kész állapotot, és ezek alapján színezi/halványítja a kártyát. Beköti a törlés és a kattintás eseményeket a kapott callbackekhez.
- **getItemCount(): Int**Visszaadja, hány listaelem (kártya) jelenjen meg a RecyclerView-ban az aktuális adathalmaz alapján.
- **setData(newLists: List)** Frissíti az adatkészletet és rendezi: sürgős/lejárt listák elöl, kész listák hátul, azon belül határidő szerint. A változás után notifyDataSetChanged()-del újrarajzoltatja a listát.

```
fun setData(newLists: List<Whislist>) {
    // Rendezés javítva: Sürgős/Lejárt legelőre, utána üres, végül a kész listák
    this.lists = newLists.sortedWith(
        comparator = compareByDescending<Whislist> {
            // Segédfüggvény nélkül is: aki sürgős, az kerül előre
            isUrgentLogic( list = it)
        }.thenBy {
            it.totalItems > 0 && it.boughtItems == it.totalItems // Készek hátra
        }.thenBy {
            it.dueDate
        }
    )
    notifyDataSetChanged()
}
```

- **isUrgentLogic(list: Whislist): Boolean**Segédfüggvény, amely true-t ad vissza, ha a lista határideje lejárt, vagy 3 napon belül esedékes (és nem „Nincs határidő”).

Az adapter a listákat **állapot és határidő** szerint színezi/priorizálja (sürgős/lejárt, üres, kész, folyamatban), és ennek megfelelően rendezi is.

További komponensek:

- Layoutok (res/layout) activity_main.xml: Főképernyő layout. item_list.xml: Lista kártya. item_shopping.xml: Listaelem kártya.
- Erőforrások (res/values) colors.xml, themes.xml, strings.xml: Színek, témák, lokalizált szövegek. Ez nincs kihasználva.
- XML szabályok (res/xml) backup_rules.xml, data_extraction_rules.xml: Mentési és adatkinyerési szabályok.
- Gradle konfiguráció build.gradle.kts, libs.versions.toml: Függőségek és projektbeállítások.
- Adatfolyam (User Flow) MainActivity → ListingAdapter → ItemsActivity → DAO → DB → UI frissítés LiveData/Flow segítségével.

Fejlesztési lehetőségek

1. **Magyar nyelv támogatása az inputoknál**
   Ellenőrizni kell az inputType beállításokat, hogy az ékezetes karakterek és magyar billentyűzet teljesen működjenek az űrlapoknál.

2. **Kihasználatlan tulajdonságok kezelése**
   Az ListItem entitásban van category mező, de nem használjuk. Érdemes megjeleníteni a kategóriát a kártyákon, és akár szűrési lehetőséget is adni rá.

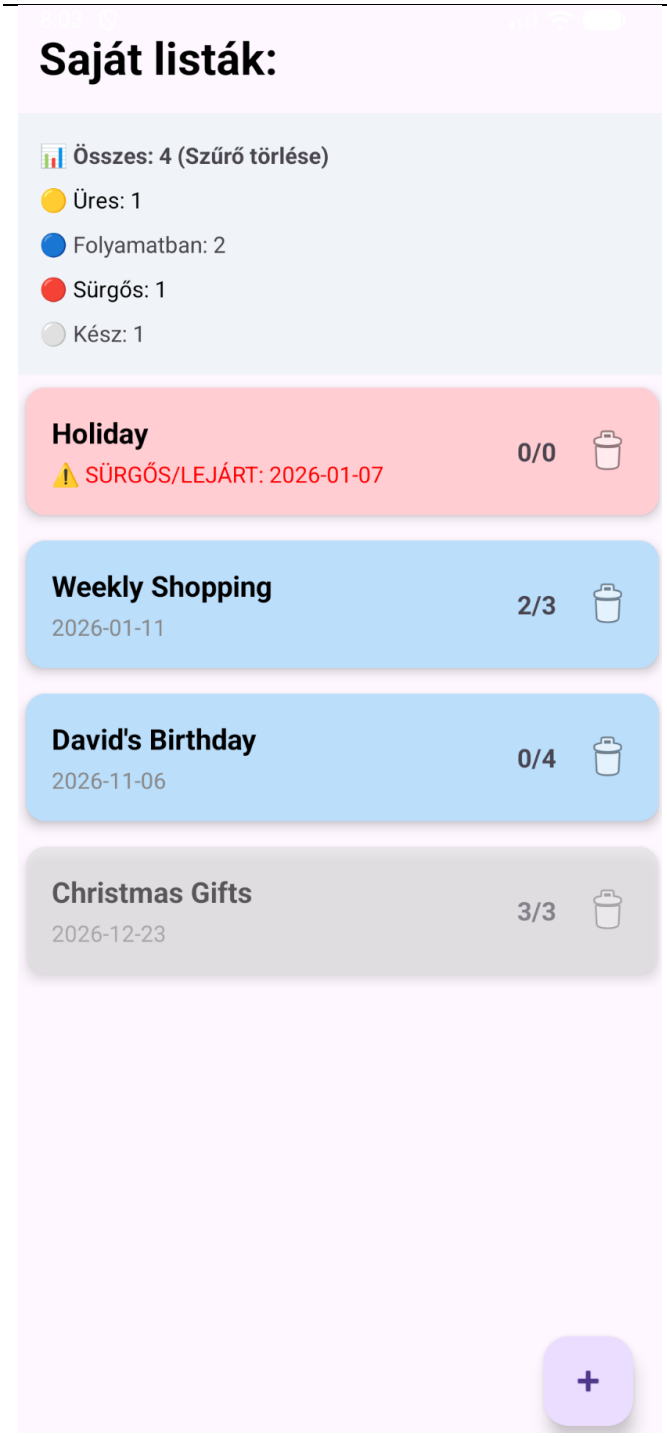
3. **Határidő formátum és megjelenítés javítása**
   A határidő jelenleg nyers formátumban (yyyy-MM-dd) látszik. Felhasználóbarátabb lenne „Ma", „Holnap", „3 nap múlva" vagy magyar dátumformátum (pl. 2026. január 6.).

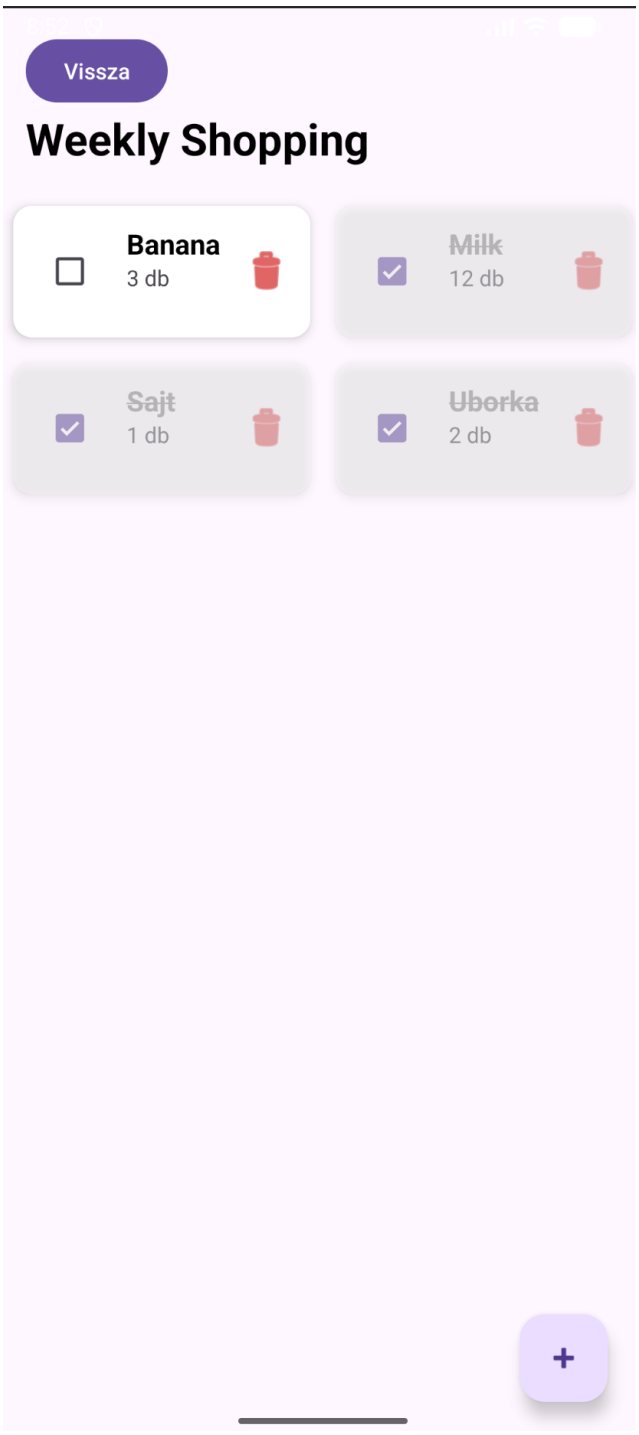
4. **Dialógusok felhasználóbarátabbá tétele**
   Az új lista és új tétel hozzáadásakor nincs hibaüzenet, ha a név üres. Adjunk hozzá egyszerű validációt és figyelmeztetést („A név megadása kötelező!").

5. **Kész listák kezelése**
   A kész listák jelenleg csak szürkével jelöltek. Érdemes lehet egy „Archiválás" funkció, hogy ne zavarják a főképernyőt, de később visszanézhetők legyenek.

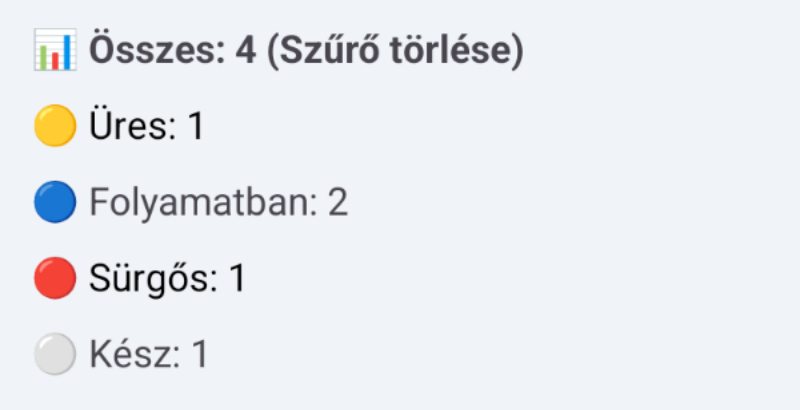
6. **Kategória és mennyiségi egység ellenőrzése**
   Az új tétel hozzáadásakor az egység (unit) mező alapértelmezett „db", de nincs ellenőrzés. Érdemes alapértelmezett értéket adni és validálni, hogy ne maradjon üres.

7. **Felhasználói élmény apró javításai**
   Animációk a lista frissítésekor (pl. új elem hozzáadásakor), visszajelzés törlés után („Tétel törölve"), és sötét mód támogatása.

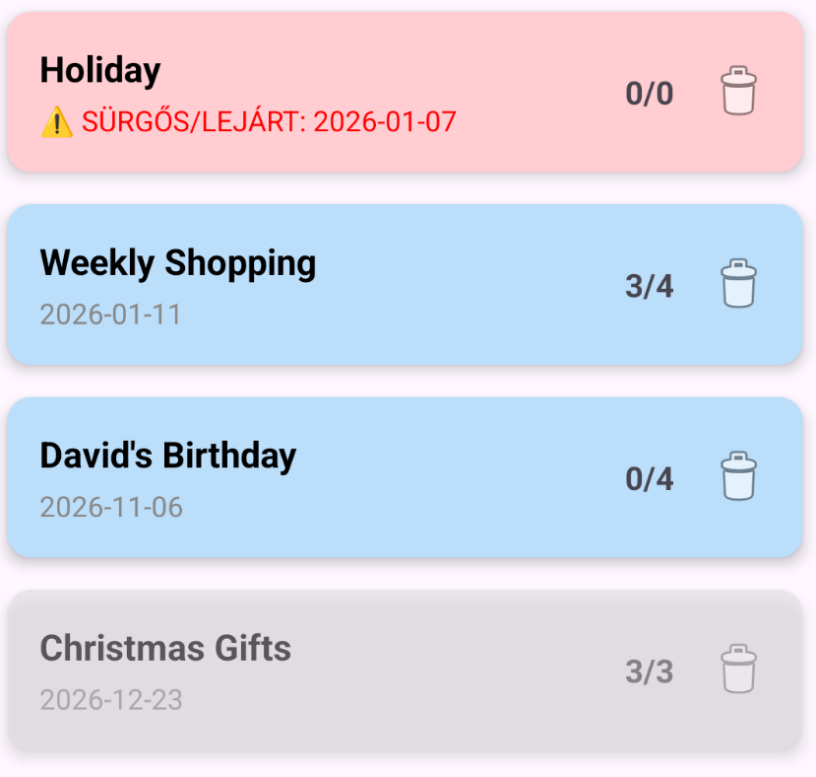




Összes kívánságlita áttekintése, villámstatisztika, koppintásra az adott kategóriára szűri a listát:



Kategóriánkénti színkódolt, rendezés és megjelenítés:



Sürgős: piros szinű, legfelül
Folyamatban: kék színű due date szerint rendezve
Kész: Szürke szinűek

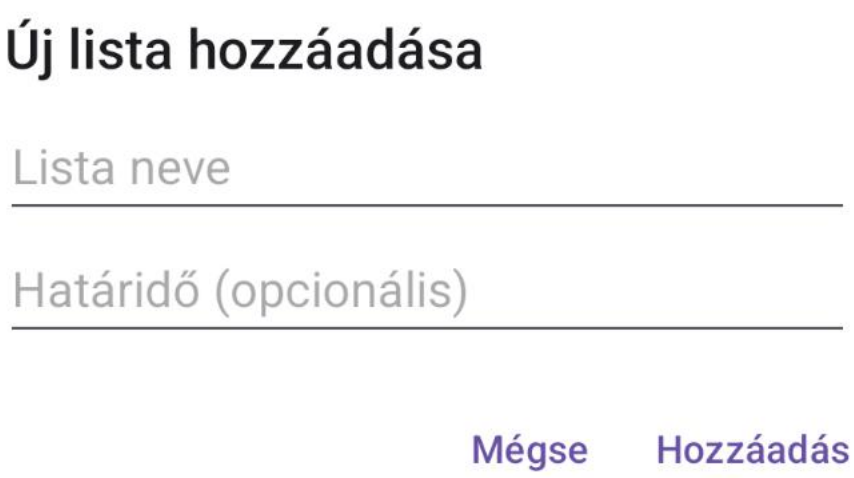
A listákat bemutató kártyákon tovább szerepel a céldátum a teljesített/teljes elemszám is.

Listák törlése:

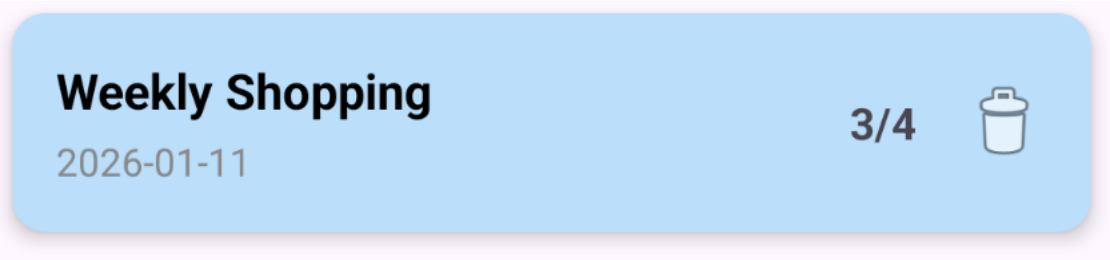
Kuka ikonra kattintva törölhetőek is a listák (ekkor minden hozzájuk tartozó elem törlődik)
A befejezett listák maguktól nem törlődnek ki!Ha visszaállítunk egy elemet, a lista státusza folyamatban lessz!

Új kivánságlista hozzáadása:

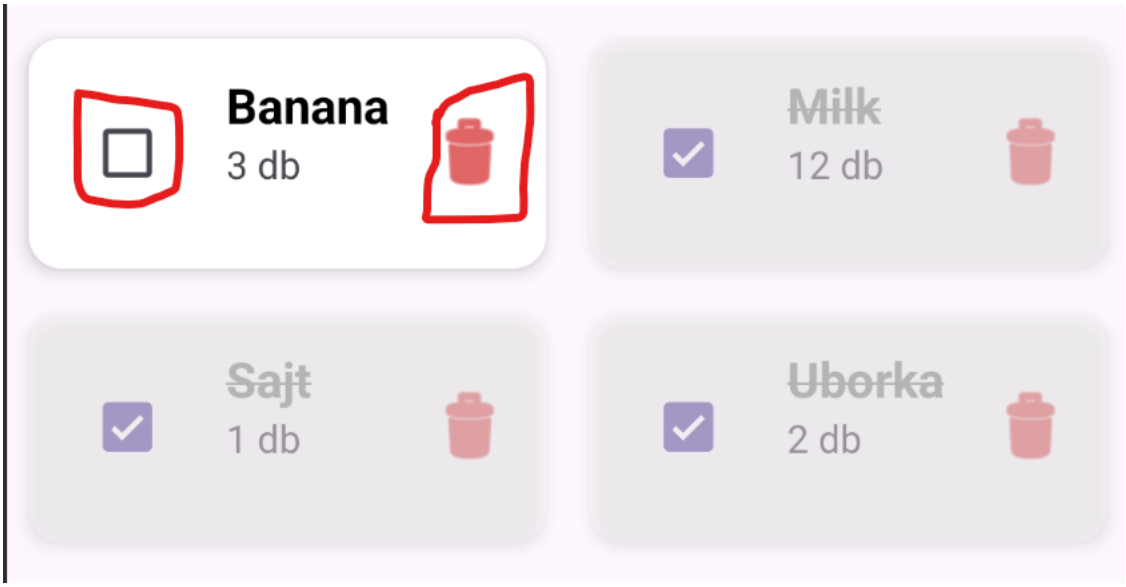
Majd felugró ablakban megjelenik az adatbekérő ablak.



Kívánságlista elemeinek a megtekintése: koppintás a kívánt listát tartalmazó kártyára:

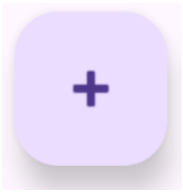


Kívánságlista elemeinek a szerkesztése



- Checkbox : az elem kész állapotba kerül, a kártya kinézete is megváltozik és a rendezésbe alulra kerül
- Kuka ikon: Lista elem törlése
- Kártyára koppintás: elem adatainak a szerkesztése

Új kívánság lista elem hozzáadása:



Adatbekérő űrlap: